ExSciTecH: Expanding Volunteer Computing to Explore Science, Technology, and Health

M. Matheny[1], S. Schlachter[1], L.M. Crouse[2], E.T. Kimmel[2], T. Estrada[1], M. Schumann[3], R. Armen[3], G. Zoppetti[2], and M. Taufer[1]

[1] University of Delaware
Dept. of Computer & Inf. Sciences

[2] Millersville University
Dept. of Computer Science

[3] Thomas Jefferson University School of Pharmacy
Dept. of Pharmaceutical Sciences

***Abstract*—This paper presents ExSciTecH, an NSF-funded project deploying volunteer computing (VC) systems to Explore Science, Tecenology, and Health. ExSciTecH aims at radically transforming VC systems and the volunteer's experience. To pursue this goal, ExSciTecH integrates and uses gameplay environments into BOINC, a well-known VC middleware, to involve the volunteers not only for simply donating idle cycles but also for actively participating in scientific discovery, i.e., generating new simulations side by side with the scientists. More specifically, ExSciTecH plugs into the BOINC framework extending it with two main gaming components, i.e., a learning component that includes a suite of games for training users on relevant biochemical concepts, and an engaging component that includes a suite of games to engage volunteers in drug design and scientific discovery. We assessed the impact of a first implementation of the learning game on a group of students at the University of Delaware. Our tests clearly show how ExSciTecH can generate higher levels of enthusiasm than more traditional learning tools in our students.**

I. Introduction

Volunteer Computing (VC) uses the computational resources of volunteers with Internet-connected personal computers to address fundamental problems in science. Unfortunately, the volunteers attracted are not representative of the general population, and their involvement tends to be low. It is known that most volunteers are white males that have already some familiarity with computers and their involvement is limited by the paradigm to donate their computer's idle cycles [1]. We believe that by making the VC paradigm more attractive to historic minorities in STEM (Science, Technology, Engineering, and Mathematics) fields, and enabling the active participation of volunteers in cognitive tasks beyond the passive cycle donation, we can use computing and mind power together to accelerate scientific discovery. The potential benefits to scientific discovery and the opportunity to engage a broader population motivate our work on the ExSciTecH project, an interactive, easy-to-use VC system to Explore Science, Technology, and Health, aiming at radically transform VC systems. To exploit the curiosity and intuition of a heterogeneous community of volunteers, we aim to reshape VC systems to educate the volunteers, so that they can use their new knowledge as active participants in advancing science. Docking@Home (D@H), a VC project powered by the Berkeley Open Infrastructure for Network Computing (BOINC) VC middleware which targets the design of new drugs for breast cancer and HIV through protein-ligand docking simulations, provides us with the thematic framework.

ExSciTecH does not replace the VC framework but extends it. Figure 1 shows how this extension takes place. Gameplay scenarios are built side by side to the VC mechanisms. Traditionally, scientists generate new simulations in VC projects by generating jobs; the jobs are distributed to volunteer's computers where they are executed when the computers are idle. This is also what happens in D@H (upper right corner of Figure 1), where the scientist generates jobs by generating new input files including a protein and a ligand. The protein-ligand complexes are sent to the D@H server where they are assigned to a volunteer's client which executes them and returns results (docked ligand conformations and their energy) to the server for the scientist's analysis. ExSciTecH plugs into

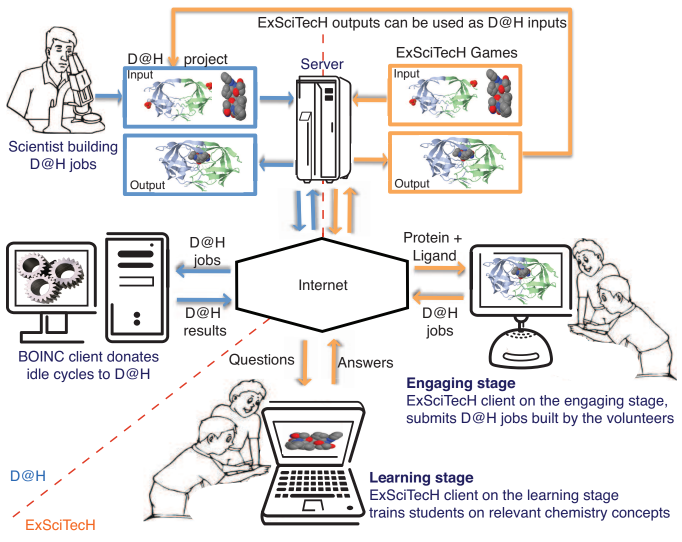

ExSciTecH outputs can be used as D@H inputs
D@H project
Server
ExSciTecH Games
Input
Output
Input
Output
Scientist building D@H jobs
D@H jobs
D@H results
Internet
Protein + Ligand
D@H jobs
BOINC client donates idle cycles to D@H
Questions
Answers
Engaging stage
ExSciTecH client on the engaging stage, submits D@H jobs built by the volunteers
Learning stage
ExSciTecH client on the learning stage trains students on relevant chemistry concepts
D@H
ExSciTecH

Fig. 1. ExSciTecH overview with the BOINC and gaming components.

the BOINC framework extending it with two main gaming components, i.e., a learning component that includes a suite of games for training users on relevant biochemical concepts, and an engaging component that includes a suite of games to engage volunteers in drug design and scientific discovery.

In this paper, we first present ExSciTecH together with the motivation for using gameplay environments to train and engage volunteers in VC projects. We then present the ExSciTecH software framework based on open-source, reusable software middleware such as BOINC [2] and tools such as VMD (Visual Molecular Dynamics) [3]. The game engine in ExSciTecH plugs into BOINC in a modular fashion. By integrating ExSciTecH into BOINC we can reuse its functionalities for other VC projects and a diverse spectrum of games beyond the games presented in this paper. Reusing codes such as VMD allows us to benefit from powerful visualization tools while supported by real scientific behavior (collisions and interactions of molecules). Finally, we present a suite of games and study the impact of our gameplay environment in facilitating learning among graduate and undergraduate students at the University of Delaware with computer science background but very little biochemistry background in a quantitative fashion. The rest of this paper is structured as follows: Section II presents the ExSciTecH's overarching goal and three main objectives. Section III described the the VC paradigm and Docking@Home, the VC project serving as thematic framework. Section IV describes the learning and engaging parts. Section V shows how we extended BOINC to integrate the game engine. Section VI discusses the quantitative learning assessment for a flashcard-like game implemented in ExSciTecH and Section VIII concludes this paper.

II. Overarching goal and objectives

ExSciTecH's overarching goal is bringing together human intuition and discernment with the unused power of millions of computers to engage diverse communities in aiding scientific discovery. Technologies that support easy-to-use and intuitive access to computers (i.e., Wiimote), intuitive graphical interfaces, and engaging gaming environments allow volunteers to contribute to scientific knowledge, specifically in the area of drug design. With the ExSciTecH interface, volunteers easily learn some basic scientific background needed to contribute to a VC project. Because of a multi-level educational approach, our system works even for those volunteers with limited scientific knowledge. With their increased knowledge, volunteers participate in advancing science by using the human intuition and intelligence that computers lack. The gaming environment allows participants worldwide to either compete or collaborate in teams to execute those complex activities in drug design that computers cannot perform as effectively and accurately as humans do. To meet the goal, we are focusing on these three objectives: (1) transform public participation in science and VC research by actively recruiting, generating and sustaining the interest of diverse populations; (2) establish self-sustaining inclusive communities of diverse volunteers engaged in computer mediated scientific investigation; and (3) increase the science delivered by volunteer computing projects to scientists in the targeted computing project by transforming participation from simply donation of idle cycles to donation of cycles plus deductive reasoning and by increasing the quantity of idle cycles donated to D@H. ExSciTecH targets three groups with different profiles and diverse demographics: (a) Group I: Graduate and undergraduate students in engineering and mathematical sciences at the University of Delaware; (b) Group II: Students with very little computer science background and those from groups underrepresented in computing, i.e., students in Womens Studies Program at the University of Delaware; and (c) Group III: Volunteers participating in D@H, our targeted computing project. In this paper we present the results for the first group; the collection of results for the other two groups is future work.

III. Targeted VC project

A. VC paradigm and Docking@Home

The VC paradigm is well established for scientific projects. On June 2012, the growing volunteer and scientist communities counted the following: 77 active scientific projects; 2,428,803 volunteers; and 7,508,886 computers distributed across the world. Berkeley Open Infrastructure for Network Computing (BOINC) [2] supports Docking@Home (D@H) [4], an NSF-funded VC project for drug design simulations. In our effort to transform the public participation in science and VC research to advance science, we use D@H as a thematic framework for ExSciTecH. D@H computationally searches for potential drugs against diseases such as breast cancer and HIV. The search is performed by simulating the docking of small molecules (i.e., ligand candidates) into proteins involved in the disease process. The ligands function as a drug and D@H aims to predict the geometry of the protein-ligand complex interaction [5]. Figure 1 shows the key components of D@H (upper left corner). Jobs comprising of a protein and a ligand candidate (docking attempts) are submitted to the D@H server, which then distributes them to computers across the Internet. Volunteers computers perform the docking calculation and return D@H results (docked molecule in the protein binding site and the resulting energies). The unknown initial conditions for the docking algorithm (e.g., initial velocity, type and position of the ligand candidates, binding site in the protein) result in a large-scale search space. Currently, over 30,000 volunteers (and more than 75,000 computers) worldwide support D@H.

B. D@H targeted diseases and scientific goals

The two proteins used in D@H as the targets for the search of new drugs are trypsin and HIV. These proteins have been selected because they are related to two high-profile diseases that are of special interest to populations underrepresented in computing: breast cancer and AIDS. Trypsin is a protease that breaks down other proteins in the digestive system. Recent studies suggest that inhibitors of trypsin can have potential application in breast cancer treatment [6]. HIV protease (HIV PR) is a protein in the HIV virus that is essential for its replication in human cells. During the process of building a new HIV virus inside the human cell, HIV PR cleaves some newly synthesized viral protein in a particular fashion. The cleaved pieces are required to build a mature HIV virus. By binding into the active site of HIV PR, a drug can eventually stop the replication of HIV virus in human cells [7].

In previous work [8], [4], we validated the prediction accuracy of two main docking algorithms by sampling between 15,000 and 30,000 docking results per day, depending on the complexity of protein and ligand, for our two proteins and for over 70 different ligands (candidate drugs). This sums up to a total of 8M hours of computation, or about 1,000 years. This very large sampling was performed at a very low cost and was possible just within 154 days (about 5 months) because of the volunteer idle cycles. On a dedicated cluster of 1,000 nodes we would have needed one whole year of computation for the same results, besides paying the high maintenance cost for the cluster. With ExSciTecH, we want now to purse a more active volunteers participation in the D@H project. The phase in which we particularly benefit from volunteers intuition and their active participation is the cross-docking phase. In the cross-docking phase, we access a very large database of ligands that have different degrees of docking affinity with a large number of closely related protein targets. One of the goals of the cross-docking phase is to identify drug side effects that result when ligands dock in the wrong off-target protein, interfering with and compromising the proteins normal function. In a naïve approach to identify side effects, we might try all ligands, with all their possible off-target proteins, and all the possible protein docking pockets. On a dedicated cluster, this search clearly can take years even with medium sized libraries of compounds ($>$10,000). Volunteers can narrow down the number of attempts in different ways. For example, guided by suggestions based on binding site similarity metrics, volunteers may investigate/assess docking pockets where ligands may dock favorably, possibly leading to off-target side effect. Also, on a different note, by selecting ligands that appear more likely to dock into protein pockets, volunteers may contribute to the reduction in the number of ligands of interest for a given protein. In all these cases, by synergistically working together with their intuition and intelligence, humans can participate and inform the searches that are ultimately evaluated by the docking algorithm.

IV. Gaming as a tool to learn and engage

A. *Motivation for using gaming*

To facilitate the active involvement of volunteers in D@H and assure their continued donation of idle cycles, ExSciTecH includes a two-stage gaming environment comprised of (1) learning and (2) engaging. The two components are showed in Figure 1 (low right corner). Although we focus on protein-ligand docking and D@H, our approach can be easily adapted to work with other VC projects. The benefits of integrating games into the volunteer experience are twofold. First, since we educate the volunteers based on their assessed knowledge, we can more effectively increase their understanding of the computational procedure. Then, our volunteers can guide the simulation process, allowing D@H to converge to more optimal solutions in shorter time periods. Second, trained, engaged, and motivated volunteers are more willing to donate their resources (computational and intellectual) to VC projects.

B. *Learning and engaging*

Each stage of our casual gaming environment is geared toward a specific goal and contains several games to achieve that goal. The first stage assesses the volunteer's knowledge about the science and teaches the volunteers about the underpinnings of the scientific problems, goals, and activities (*learning stage* in Figure 1). More specifically, the learning stage evaluates the volunteers knowledge with respect to computer science and its application to scientific problems, specifically protein-ligand docking. In the second stage, volunteers engage in our VC project by playing with a suite of games directed to spark their interest in science and computing in general and in our D@H project in particular. In this stage participants contribute their mind-power as well as their computing-power to D@H scientific goals (*engaging stage* in Figure 1).

In the *learning stage*, volunteers become players and we assess their current level of understanding before tailoring the engaging stage to meet their specific needs and skills. When players complete the learning stage, they can earn a rank of Novice, Amateur, or Professional Chemist, based on their performance. To this end, we are designing several game modules that test both factual knowledge as well as visual/spatial adeptness. The first learning game we have developed and we use in this paper is a molecule flashcards game. Players are presented with the 3D representations of a selected dataset of biochemical molecules. As each molecular conformation slowly floats from the top to the bottom and then off of the screen, players are asked to identify or categorize each floating conformation based on key properties. The structures fall one at the time at different speeds, the player can rotate and analyze them then attempt to classify each by identifying distinguished properties. Different levels of complexity have been implemented, from the easier levels requiring the differentiation among e.g., atoms, peptides, and proteins, to more complex levels including the identification of particular molecules (e.g., vitamin A, vitamin E) or the identification of molecules present in different types of food. Graphical clues that are embedded in the environment guide the players toward correct answers; text is only used for minor directions and for naming. If players get an incorrect answer, they are further presented with hints, the helpfulness of which is dependent on the player's level of expertise. Figure 2 shows a snapshot of the flashcard game in which the player has to indicate whether the visualized molecule is a lipid, peptide, carbohydrate, or nucleotide. In the same figure we can see the additional information related to the answer, the score, and a hint for further discovering.

In the *engaging stage*, the assessed players can further study science in D@H by using their personal knowledge and intuition. Specifically, the goal of this stage is to encourage

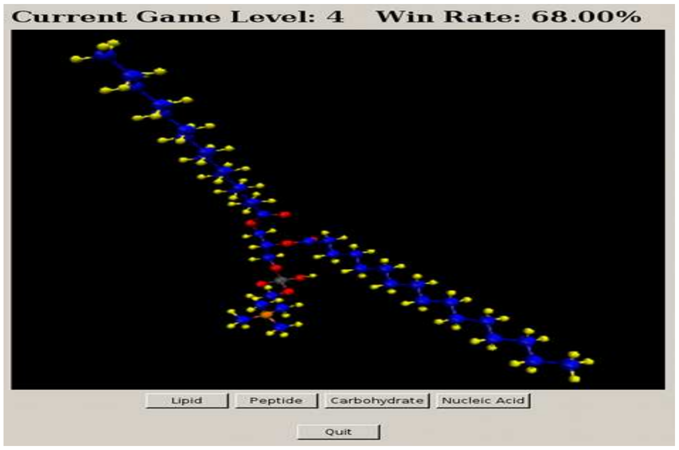


Fig. 2. A snapshot of the molecule flashcard game.

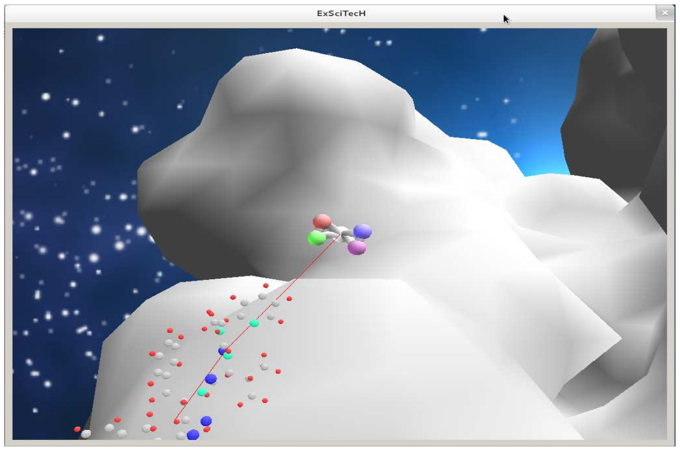


Fig. 3. Snapshot of the Drag'n Dock game.

players to use the training from the previous stage as a guide to generate and distribute useful computation through D@H. At this stage, players can choose the value of one or more features of a D@H job (e.g., the protein-ligand pairs, the initial atomic velocities, and the binding sites). The players can combine the selected features and their values to generate one or more jobs and submit them to the D@H server, where the submissions are queued to be distributed to other players as well as volunteers not participating in the game, together with the scientist's jobs. The engaging stage differs from the previous stages because the player can freely choose job conformations and setting and then she can submit this initial information to the D@H server becoming a de facto scientist, while competing with other players and the D@H scientists. For example, the player could choose to design a new ligand candidate by selecting and modifying the structure of an existing ligand. By using the player's intuition, we aim to reduce the cost of the stochastic search space and allow volunteers to directly participate in the development of new drugs. The first engaging game that we designed and we are currently integrating in BOINC is called "Drag'n Dock". When a player starts this game, she is presented with a protein that she rotates to identify the binding site. The player also chooses a drug (ligand) to dock from a set of possible drugs. The player then flies a spaceship towing the ligand into the protein's binding site in an attempt to dock the ligand. Once the ligand has been docked, the player flies their spaceship through a portal to tell the game to submit a job to the D@H server. Once the job has been submitted to the server, it will be executed on another volunteer's computer. Figure 3 shows a snapshot of the "Drag'n Dock". ExSciTecH keeps track of the outcomes of the engaging modules so that both the players and the scientists can monitor progress at runtime by accessing the D@H Webpage. We also developed a reward system based on these outcomes (i.e., final energy and overall quality of the volunteers results) as well as the total donated computation time. Initially each player is assigned with a certain amount of computing time that can increase or decrease based on the player scores and participation in donating idle cycles. In other words, we can reward players with further docking attempts (playing time) either because they submit scientifically relevant results or because they complete jobs from other players. All these mechanics boost the volunteers' motivation both in donating idle cycles and in exploring science in D@H.

V. Integrating Games into BOINC

One key aspect of ExSciTecH is the re-usability of open-source software to build its gaming environment. This includes extending the BOINC framework to integrate our gaming engine and the associated games as well us using a visualization tool like VMD to provide the 3D scientific representation of molecules in the games. Figure 4 shows the original BOINC framework and the new ExSciTecH parts together with their interactions. In Figure 4, we separate the client and server

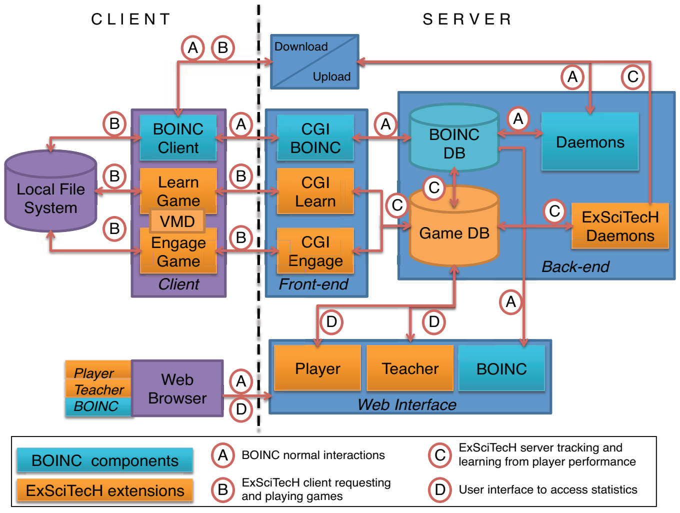


Fig. 4. An overview of BOINC with ExSciTecH integrated.

sides by a dotted line. In the server, we show the *front-end*, *back-end*, and *Webpage interface* as blue containers.

In the client, we show the augmented *client* software as a purple container. Within the client and server ”containers” the different software parts are shown as blocks while interactions are shown as arrows. Original BOINC components are shown in bright blue and ExSciTecH components in orange. The interactions between components can be classified as (A) original BOINC interactions – these are interactions that are in the original BOINC software; (B) ExSciTecH game requests – these interactions refer to the ExSciTecH client issuing requests and receiving replies to and from the server front-end; (C) ExSciTecH server tracking and learning from player performance – these interactions refer to those between the front-end and back-end to keep track of player's progress and define strategies to better aid in the learning and engaging processes; finally (D) user interface to access statistics – they refers to the interactions between the Webpage interface and the databases to show statistics and to customize games packages for the specific player's skills. The several parts are discussed in detail in this section.

A. Original BOINC framework

The BOINC software is based on a master-worker topology and has two main parts: the server side and the client side. The server is usually adapted for a specific VC project by customizing some framework daemons. Every project usually has several thousands of clients donating computing power when idle. The *server* generates jobs, assigns jobs to clients, and finally collects and validates results when they are returned by clients. A BOINC server is centered on a relational database (*BOINC database*), whose tables correspond to the abstractions of the BOINC computing model, i.e., the description of attached clients, the single or multiple applications or programs in a project, and the jobs associated with a project. The BOINC server relies on a set of daemons (*BOINC daemons*) to generate new jobs (workgenerator), to periodically populate a shared memory segment with jobs taken from the BOINC database (feeder), to change the status of jobs in the database, e.g., unsent, in progress, over, success (transitioner), to validate results (validator), and to remove files associated with jobs that have been validated or that are no longer needed (assimilator/file-deleter). When the BOINC server receives a request for jobs from a client, a *CGI script* serving as scheduler validates the authenticity of the client and, if available, assigns one or multiple jobs to the client in an XML reply message. Additionally to the front-end (CGI script) and back-end (BOINC daemons and database), BOINC has an administrative *Webpage interface* that project administrators use to access and administer the BOINC database, and BOINC volunteers use to follow up their statistics and jobs status as well as credits rewarded.

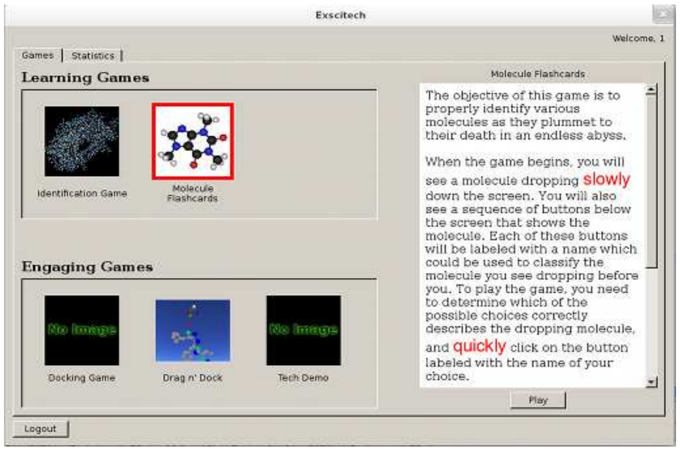
Clients enable volunteers to donate their computer's idle CPU cycles to advance science. Volunteers download the client software and attach themselves to one or more BOINC projects. The client runs in the background and goes to sleep while the volunteer's computer is in use. Once the idle CPU percentage goes above a user-defined threshold, BOINC resumes execution. The client is in charge of contacting the server to request jobs and retrieve their associated input files. When the client gets new jobs, these jobs are executed in a round robin manner, sharing the client's CPU evenly among the several projects to which the client is attached to. After the execution of jobs is completed, the client contacts the server to return job results and upload their respective output files. When a BOINC client is attached to a project, it periodically issues an XML scheduler request to the project's server. The request message includes the description of the host and its current workload (jobs queued and in progress), the description of newly-completed jobs, and the request for new jobs.

B. BOINC's extensions to integrate ExSciTecH

An important aspect of the ExSciTecH gaming engine is that it is integrated into the actual BOINC framework (as shown in Figure 4) and does not substitute it. In particular we maintained the BOINC communication protocol between client and server unchanged and expanded the software parts of both server and clients in a modular fashion. In this way we can take advantage of the widely used VC paradigm for different projects beyond D@H and for different games beyond the ones proposed in this paper.

At the *server level*, we augmented a BOINC server with ExSciTecH extensions to enable the generation, execution, tracking, and scoring of a diverse suite of games, similarly as original jobs are generated, executed, tracked, and scored in BOINC projects. The server extensions include a database (*Game database*), new CGI scripts, and new ExSciTecH daemons. The game database contains the game information. There are a number of tables in the database which are general to ExSciTecH such as the game tables, which keep track of games and game sessions similarly as the project and application tables keeps track of each project and its applications in the original BOINC projects, and the player table, which is a parallel table to the BOINC user table and includes game records for each player. This database also contains tables that store relevant data on the different molecules used in our games; these tables are game specific similarly as for the project-specific tables in any BOINC project. In ExSciTecH, these tables include data on molecule name, type, and other properties as well as references to their atomic and graphic representation. The game requests consist of very small and frequent messages. At the same time, the existing BOINC CGI scripts already have high overhead. Thus, instead of using the BOINC CGI scripts for game requests, we designed a custom CGI script for game requests and implemented the script in Python. Our game-request CGI script interacts with the game database as the BOINC CGI scripts interact with the BOINC database. As for the BOINC CGI, our CGI receives requests in an XML format. In the XML file, there are general tags used for any type of games; they are `user_id`, `request_id` and `wu_name`. The `user_id` and `wu_name` fields are used to identify the game session while the `request_id` tells the script how to handle the request. The request can have additional fields that are passed to the request handler associated with the `request_id`. To add a new game with server-side functionality, we just need to define a new `request_id` and a new handler for the `request_id` in the script. A subset of the XML tags is game-specific. For example, in the molecule flashcards game, each request through the CGI scripts has two additional fields, `pdb_name` and `answer`. The `pdb_name` tag is an identifier for what question the user is answering and the `answer` tag is the answer the user is submitting. The server uses the session information and the question identifier to check the correctness, then it identifies what questions the user has left to answer, picks one, and replies by using a formatted XML. For the generation and distribution of game sessions as well as for the collection of game results, we add game-specific daemons to the existing BOINC daemons. In particular ExSciTecH needs custom *workgenerator* and *validator* daemons for each game. The workgenerator is in charge of building packages of molecules to be used as input for a given game. The validator is in charge of assessing the validity of a game and moving the obtained score from the game database to the BOINC database in the form of ”credit”. Credits are greatly valued by the volunteers and are virtual units acknowledging one’s contribution to the BOINC project proportionally. In this way, the volunteer’s participation in one of our games is additionally rewarded in the form of project credit. Credit transfer is also possible from the BOINC database to the game database when a volunteer finds a scientifically meaningful result (e.g., correct protein-ligand conformation). In this case, the credit assigned to the BOINC job is transferred to the game job and can eventually be used to increase the proficiency level of the player or submit additional game jobs. The addition of daemons to coordinate the interaction of BOINC with the games helps in keeping the encapsulation of tasks and the reliability of our software. This is because the new daemons are the only ones interacting with both the BOINC database and the game database directly.

At the *client level*, we added a gaming part to the BOINC client that works in concert with it. When players start the client, they are presented with a series of games they can play (Figure 5). The games are separated into two categories: learning games and engaging games. The game client, like other components in this project, was designed to make it easy to expand and add additional games. Another important aspect of the client software re-usability is the use of VMD (Visual Molecular Dynamics)to display 3D molecules [3]. The 3D representations help give players a better understanding of the molecules and add a certain level of interest to the games. To this end, we use VMD as the rendering engine for our 3D graphics. VMD is a molecular visualization program for displaying, animating, and analyzing large biomolecular systems using 3D graphics and built-in scripting. We use portions of the VMD source code for rendering molecules. Whenever a game is started, VMD runs in the background and provides our games with its robust molecular rendering engine as well as a reliable scientific understanding of molecular behavior (e.g., collisions and reactions). Last but not least,



The objective of this game is to properly identify various molecules as they plummet to their death in an endless abyss.

When the game begins, you will see a molecule dropping slowly down the screen. You will also see a sequence of buttons below the screen that shows the molecule. Each of these buttons will be labeled with a name which could be used to classify the molecule you see dropping before you. To play the game, you need to determine which of the possible choices correctly describes the dropping molecule, and quickly click on the button labeled with the name of your choice.

Fig. 5. The ExSciTecH game client.

we added two new PHP interfaces to the BOINC framework, one for students and one for teachers. In the student interface, players can view their standings and progress in different games. In the teacher interface, teachers are able to customize some of the games for use in their classes. For example, in the molecule flashcards game, teachers can build custom sets of flashcards to give to their students that focuses on material specific to their class.

VI. Testing

A. Experimental setting

The main goal of this work is to assess the impact of the gameplay environment on the learning process compared with more traditional learning approaches such as the use of paper flashcards. For a quantitative and rigorous assessment, we engaged 24 computer science students: 14 undergraduate students and 10 graduate level students with little knowledge in drug design. We gave a 10-minute, easy-to-follow introduction to all the participating students; the introduction was on key aspects of molecule properties i.e., if a molecule is a peptide, carbohydrates, lipids, or nucleotide. After the short introduction, we immediately equally split the students in two groups. The first group was asked to take a test identifying 2D diagrams of 45 molecules as either peptides, nucleotides, carbohydrates, or lipids. They had to work individually, were given 20 minutes to complete the test, and had access to a computer for searching. The time for the test on paper was chosen because it is also the maximum time needed to complete a round on the molecule flash cards game. The second group was asked to identify the same properties from the same set of molecules by using our game. We arranged computers in the department where the students could play. To preserve the privacy of the students, each student participating in the game was assigned with an individual, random code so that the student’s name never appeared in any form or any file.

B. Results and discussion

We compared the two groups of students in terms of three metrics: percentage of correct answers (Score); level of enjoyment in a scale from 1 to 10 (Enjoyment), with 1 being no fun and 10 a lot of fun; and total time spent answering the 45 questions (Time). The label "Game" identifies the group of students using our learning game and the label "Paper" identifies the group of students using a color printout. The box-and-whisker diagram in Figure 6 shows lower and upper quartiles at the top and bottom boundaries of the box; the median is the band inside the box; the mean is the black square; the whiskers extend to the most extreme data points or outliers; and outliers are plotted individually as + symbols. If we compare the scores for the two groups, we observe that

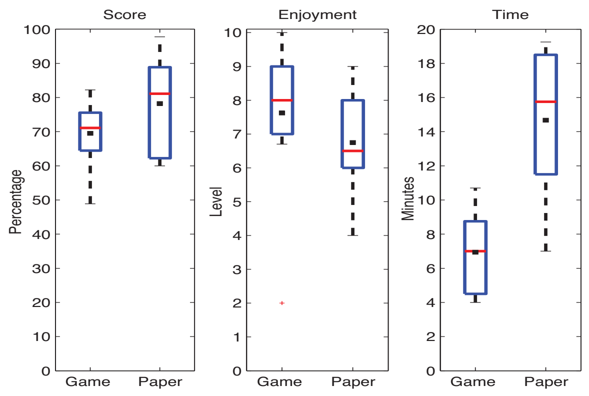


Fig. 6. Box-and-whisker diagrams for score, enjoyment, and time.

on average the Paper group outperforms the Game group by 10%. On the other hand, if we compare the perceived level of enjoyment, the Game players were more engaged than the Paper players with a median enjoyment of 8 compared to a median enjoyment of 6.5. In addition, it is worth noting that the Game group had a more consistent perceived level of enjoyment than the Paper group, since the latter showed a larger spread in their level of enjoyment. If we compare the time each of the two groups took to answer all 45 questions, the difference is even larger than in the previous two metrics. On average the students playing our game answered the 45 questions in 7 minutes, that is 10 seconds per question, while the Paper group took twice as long on average. We think that the serial nature of the game questions and the time limits imposed by the individual questions prevented participants using the game from doubling back to double-check their work and from staying too long on the individual problems. The falling molecules might have contributed to a more rushed atmosphere for the people playing the game, speeding up their responses and possibly causing increased error. To gain further insight into why the Game group performed unfavorably in terms of correct answers, we analyzed the percentage of mistakes for every question. Figure 7 shows the difference between the percentage of students that missed a question in the test and the percentage of students that missed it in the game for each of the 45 questions (x-axis). There appears to be a steep learning curve at the beginning of the game as the number of missing and wrong answers was highly concentrated in the first 8 questions (see the vertical dotted

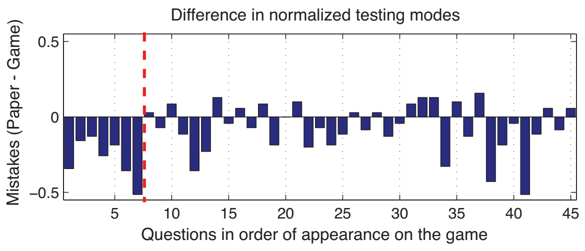


Fig. 7. Errors in paper flashcards vs. game flashcards.

line); after this learning phase, the errors seem to be more related to the difficulty of identifying the molecules. Even taking this initial learning period into account, students taking the paper test still appear to have performed with a higher level of accuracy. Several factors might have contributed to this result. Firstly, certain questions were much easier to identify on the paper test as their 2D structures made certain features readily identifiable (e.g., peptide bonds, phosphate groups), while their 3D conformations were far more confusing. For instance, question 41 consisted of a large circularly linked peptide with many distinct peptide bonds. In the test, these peptide bonds were readily apparent as they looked exactly like the peptide bonds in the handout given to the students. The 3D conformation of this molecule was less orderly, making it hard to identify these bonds and therefore making the identification more difficult. Secondly, game players might have felt rushed as the falling molecules put a limit on the amount of time they could spend on the individual questions, a limitation the paper test takers lacked. Thirdly, game players could not double check their work. Once a molecule had been identified they could not go back and change their answers as new information became available. And fourthly, double bonds were displayed as single bonds in the game (the visualization of double bonds is work in progress) which appeared to confuse several of the students from the feedback we received. In contrast, double bonds were accurately represented on the paper test. To address these issues, we will develop mini-tutorials to train the users to interact with the game before testing. We will also make the molecule's dropping speed dependent on the player's level of success (i.e., molecules slow down when the player makes frequent mistakes). Finally, we plan on implementing better visual clues such as labeling atoms and showing double bonds.

VII. Related work

Other initiatives target VC projects and their communities. Fold.it [9] is a well-known project targeting challenges in molecular biology. Developed by the Rosetta@Home team, Fold.it aims to use a volunteers intuition and competitiveness to fold proteins through puzzles. It has no formal assessment and education components, although learning may occur as a byproduct. Contrary to Fold.it, our system uses an alternative, more intuitive human-computer interface (i.e., the Wiimote) and includes game modules specifically to emphasize assessment and learning components. Because of this emphasis, our project can target a broader community, including beginners in the field. Although we enjoy Fold.it, we still find that Fold.it is non-trivial for novices in the field. The BOINC team has developed Bossa [10]. Bossa is an open-source software framework for distributed thinking that targets the use of volunteers on the Internet to perform tasks that use human cognition, knowledge, and intelligence. Similarly to Bossa, we are targeting the volunteers knowledge and intelligence; however, we also rely on intuitive and engaging gaming environments to first train our volunteers how to become educated, active contributors to science. Cusack et al. [11], [12], [13] described a lack of broader appeal in current VC projects that limits the VC demographic to technologically savvy volunteers. In [11], [12], [13], the authors proposed casual gaming as a method to overcome these limitations by outlining the importance casual gaming has in attracting a more diverse community of volunteers. We find this work both apt and inspiring, and thus we use it as a foundation for our own contributions. For example, the authors present the Wildfire Willy game, in which players protect a forest from a raging fire. While the authors admit that players are not really contributing anything meaningful to the computations since they have no way of making intelligent choices, they believe the greatest potential lies with games that combine human insight with raw computing power. Our goal is in fact to go beyond the theoretical proofs presented by the authors and provide a casual gaming experience in which volunteers can contribute meaningful insight to the computational process. The work also discusses game design principles important to the success of a volunteer game. We leverage these principles in developing game modules that not only appeal to a wide demographic of volunteers, but also assess and teach them to become informed, meaningful contributors to D@H.

Outside the VC community, other work targets human computation by harnessing the combined computational power of humans and computers to solve large-scale problems. Luis von Ahn's work at Carnegie Mellon University inspired our work [14]. We agree with von Ahn that games with a purpose can be harnessed to solve difficult computational problems. Our purpose is to teach science to public participants in VC projects so that they can actively contribute to research activities. Protein-ligand docking exemplifies a large-scale, open problem that can be solved using docking algorithms that may be directed from the influence of collective human brain power. By training volunteers with varying degrees of skill at configuring docking searches and using the Internet, they can perform the computation as suggested by von Ahn. Our gaming environment is designed so that the players, once trained by using incentives, can perform searches for the scientists that may be useful. Also, vendors such as Amazon are using similar principles in their Mechanical Turk initiative. Contrary to our work, Mechanical Turk uses the public but does not aim to educate and involve them in the learning process.

VIII. Conclusion and future work

In this paper we presented ExSciTecH, a software framework that, by plugging its game engine into BOINC, can transform the way volunteers participate into VC projects from passive (i.e., simple donation of idle cycles) to active (i.e., exploration and generation of new scientific simulations). The overall ExSciTecH framework consists of a two-stage gaming environment that comprises learning and engaging games. We show how tests on a sample of students at the University of Delaware outlined the higher level of enthusiasm for the students when using ExSciTecH rather than traditional learning tools (e.g., paper flashcards). At the same time observing the players helped us to identify new directions and improvements for ExSciTecH, e.g., variable speeds of the games based on the level of successes of the player and use of mini-tutorials on how the player can get confident using the game interface.

Acknowledgements

M.M. and S.S. have contributed equally to this work. This work was supported by the NSF IIS #0968350/#0968368 entitled Collaborative Research: SoCS - ExSciTecH: An Interactive, Easy-to-Use Volunteer Computing System to Explore Science, Technology, and Health. The authors thank John E. Stone (UIUC) for the VMD support and the D@H volunteers for providing us with essential resources for our docking simulations.

References

[1] T. Estrada, et al. Benchmarking Gender Differences in Voluntary Computer Projects. Poster in Proc. of the 2012 Grace Hopper Celebration of Women in Computing (GHC12), 2012.

[2] D.P. Anderson. BOINC: A System for Public-Resource Computing and Storage. In Proc. of 5th Int. Workshop on Grid Computing, 2004.

[3] W. Humphrey, A. Dalke, and K. Schulten. VMD - Visual Molecular Dynamics. J. Molecular Graphics, 14, 3338, 1996.

[4] M. Taufer, R.S. Armen, J. Chen, P.J. Teller, and C.L. Brooks III: Computational Multi-Scale Modeling in Protein-Ligand Docking. IEEE Engineering in Medicine and Biology Magazine, 2008.

[5] G. Wu, D.H. Robertson, C. L. Brooks III, and M. Veith. Detailed Analysis of Grid-Based Molecular Docking: A Case Study of CDOCKER - A CHARMm-Based MD Docking Algorithm. J. Comp. Chem., 24, 15491562, 2003.

[6] J. McIlroy, B. Cullen, G. Kay, J. Nelson, W. Odlings, R. Spence, and B Walker. A Novel Trypsin-like Enzyme in Breast Cancer. Biochem. Soc. Trans., 22(1), 19S, 1994.

[7] L. Montagnier. Historical Essay: A History of HIV Discovery. Science, 298(5599), 1727 1728, 2002.

[8] M. Taufer, M. Crowley, D. Price, A. Chien, and C.L. Brooks III. Study of an Accurate and Fast Protein-Ligand Docking Algorithm based on Molecular Dynamics. Concurrency and Computation: Practice and Experience, 17(14), 16271641, 2005.

[9] D. Baker et al. Fold.it, Solve Puzzles for Science. http://fold.it/portal/, 2008.

[10] D.P. Anderson et al. Bossa, an Open-source Software Framework for Distributed Thinking. http://bossa.berkeley.edu/, 2008.

[11] C.A. Cusack, E. Peck, and M. Riolo. Volunteer Computing Games: Merging Online Casual Gaming with Volunteer Computing. In Proc. of Int. Academic Conference on Meaningful Play, 2008.

[12] E. Peck, M. Riolo, and C. Cusack. Wildfire Wally: a Volunteer Computing Game. In Proc. of 2007 Conference on Future Play Int. Conference on the Future of Game Design and Technology, 2007.

[13] C. Cusack, C. Martens, and P. Mutreja. Volunteer Computing Using Casual Games. In Proc. of 2006 Int. Conference on the Future of Game Design and Technology, 2006.

[14] L. von Ahn, S. Ginosar, M. Kedia, R. Liu, and M. Blum. Improving Accessibility of the Web with a Computer Game. In Proc. of the SIGCHI Conference on Human Factors in Computing Systems (CHI 06), 2006.